
# Forensic Analysis Reveals Data Leaks in HIPAA Compliant Software

**Megan Bell**
**Senior Director, Analytics, Kivu**

**Adam C. Solander**
**Member of the Firm, Epstein Becker Green**

***October 2015***

# Forensic Analysis Reveals Data Leaks in HIPAA Compliant Software

## Summary

The forensic analysis of a supposedly HIPAA compliant[1] electronic medical records ("EMR") software revealed various ways that the application was creating and storing unencrypted PII/PHI in undisclosed, yet accessible, locations on the Covered Entity's network. The problems were mostly due to:

I. Default settings in the application that store generated reports (containing PII/PHI content)[2] or patient-specific medical files outside the application's otherwise secure database. While ordinary users may only be able to access such generated reports or patient files by logging into the EMR application, the actual reports or patient files are stored outside the application's database and are thus accessible depending on the user rights to the hosting server;

II. Database transaction logs, which are designed to ensure database integrity and to audit HIPAA compliance, are stored in clear text outside the secure database; and

III. Users failing to delete or secure files containing patient data generated by the application or associated with use of the application, that remain accessible to other employees due to poor access controls.

## Introduction

In a recent data breach investigation, Kivu encountered an integrated EMR software solution that stored patient records including social security numbers ("SSNs") on a Windows server[3]. While the EMR application had protected access with unique credentials assigned to users, the server itself was accessible to all employees with domain credentials. The EMR software offered complete practice management capability in a single offering (such as patient management, prescriptions ordering and tracking, patient communications and billing). However, the software:

1) Failed to secure patient data from broad system access;
2) Stored files to default locations outside the secure application; and
3) Created transaction logs containing PII/PHI in clear text stored outside the secure application on the server hosting the program.

---

[1] A non-profit organization named The Certification Commission for Healthcare Information Technology *("*CCHIT"*)* was founded in 2004 in order to certify electronic health records ("EHR") and health information exchanges ("HIE"). The organization formally closed in November 2014, without transition of business or a replacement organization. (Source: https://www.cchit.org)

[2] Personally Identifiable Information ("PII") is information that could distinguish an individual. Protected Health Information ("PHI") refers to individually identifiable health information such as a current medical condition that identifies an individual or which provides a reasonable basis to identify an individual. (Source: http://www.hhs.gov/ocr/privacy/hipaa/understanding/summary/)

[3] While the specific EMR software application is not named in this report, the findings should act as guidance for organizations seeking to test the security of their own EMR applications.

## Case Summary

Kivu was retained to investigate a brute-force attack against a medical facility's EMR system. Hackers deployed automated password-guessing software to gain access to the Windows server through a Remote Desktop ("RDP")[4] software connection. Once the software successfully determined the password, a hacker manually logged into the server using the RDP connection. The primary server used for managing patient information (including billing and insurance) was compromised by hackers who gained administrative user rights over the server. Significantly, the hackers did not gain administrative rights over the EMR database hosted on the server (which had different login credentials than the server itself). Since the medical facility believed that no PHI/PII was stored on the server outside the secure EMR database, it initially believed that there was no compromise or theft of PHI/PII.

However, during the investigation, Kivu discovered a significant repository of unsecured patient data (including SSNs, credit card numbers, driver license numbers and dates of birth) that was stored in clear text outside the EMR database. Other sensitive information such as EMR system user names and passwords were also present in clear text.

The patient data was found in the EMR software's supporting folders that were stored outside of the secure database environment. Analysis identified three types of files:

1) Automatically generated logs that contained PII/PHI;
2) User generated output of the database; and
3) EMR software patient records attachments (e.g., PDF files of billing request letters).

While it is not uncommon to find correspondence and reports generated by an EMR application stored outside the application's database, the database entry logs files should have been secured and stored within the EMR's database software.

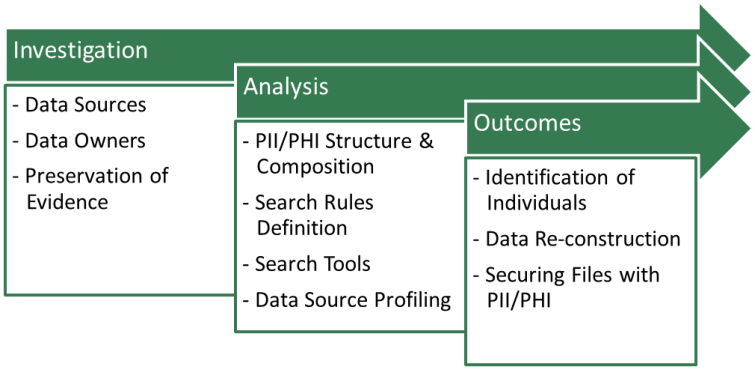
This came as an unpleasant surprise to the client, and served as a lesson on how (supposedly) HIPAA compliant software can "leak" PHI/PII — leaving it in unsecured areas on the network.

[4] Remote Desktop software is a Windows utility that connects two computers across a network. (Source: windows.microsoft.com/en-us/windows7/products/features/remote-desktop-connection)

## Analyzing Data Exposure

In a breach investigation, potential data exposure analysis begins early in the forensic investigation process. The first signs of data exposure are often identified when addressing the questions "Was there a breach?" and "How did the breach occur?" In this brute-force attack scenario, once it was determined that compromise of the server was likely, Kivu commenced an immediate high-level review for the existence of PHI/PII. When PHI/PII is identified, data exposure analysis becomes a separate component of the forensic investigation with a process that is focused on evaluating (i) the types and amount of PII/PHI and (ii) the identification of the potentially affected individuals.

**Data Exposure Analysis Process for EMR-related Data**




### 1. Investigation

**Example Finding: SSN in Log File**

```
3235 337C 3B7C 736F 6369 616C 5F73 6563  253|;|social_sec
7572 6974 793D 3636 3637 3732 3334 3520  urity=666772345
```

Re-created database transaction log entry (in hexadecimal view)

For the compromised server, Kivu discovered the first signs of PII/PHI (SSNs) in a database transaction log file[5]. The log file contained transaction events, including patient records, and was formatted in a proprietary structure. However, the log file contents were not protected, and transaction events were visible in clear text. As illustrated above, Kivu's forensic tools[6] were able to search the database log files for signs of accessible PII/PHI without the need to decode the data.

---

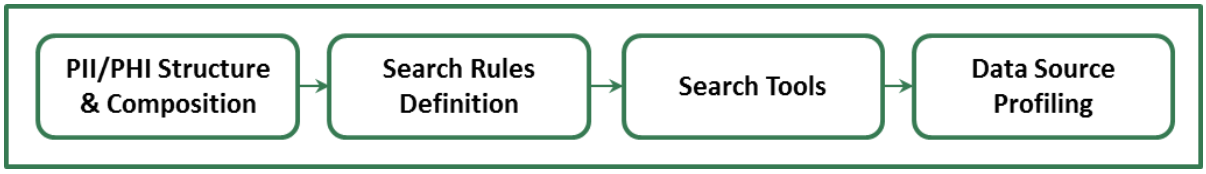
[5] A database transaction log is a history of events executed by a database system. The transaction log supports the ability trace additions, modifications, or deletions within a database. The proprietary file formats associated with database transaction logs may require the use of special software to review file contents. A database transaction log should not be confused with other file types that contain database-related data but do not support the functionality of the database. Examples of other database related files include batch exports of data for reporting, exception reports concerning data processing, or data files for the purpose of migrating data.

[6] Kivu utilizes a combination of forensic analysis technologies such as EnCase 7 and Nuix to examine hundreds of different file types ranging from simple text files such as an HTML files to complex enterprise files such as databases or email servers. Kivu also employs other tools such as regular expression utilities to search machine-level data rather through words, phrases or alphanumeric character sequences that are human recognizable. By

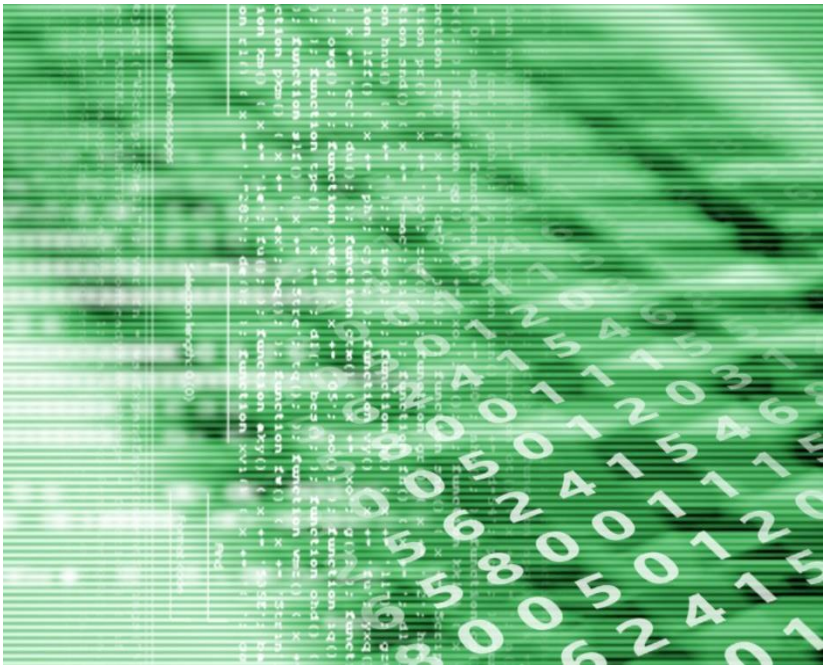
Further investigation of the log files revealed that many freely available file editing utilities, such as PSPad[7], could be used to identify and review PII/PHI without the need for specialized forensic tools or database log file viewers.

## 2. Analysis

**Components of the Search Process**




Due to the presence of searchable SSNs in the transaction logs, Kivu suspected other sources of PII/PHI related to the EMR application could be present on the compromised server. The search for PII/PHI was extended in order to examine the entire server for other potential sources of PII/PHI. Due to the size of the array of hard drives (> 5 TB), Kivu conducted narrowly defined, high-level searches to identify other potential sources of PII/PHI. The search criteria used were based on factors including types of potential PII/PHI that may be encountered in an investigation, PII/PHI associated with regulatory compliance (e.g., notifiable types of personal information under HIPAA/HiTech), relevant states' notification laws, and unique characteristics of the specific client's data, such as the data elements contained within the EMR database. The objective was to accurately identify a breadth of potential PII/PHI for further review while reducing the number of false positives.



There are four components to achieving the successful identification of PII/PHI across a large structured or unstructured data set.

i. PII/PHI Structure and Composition
ii. Search Rule(s) Definition
iii. Search Tools
iv. Data Source Profiling

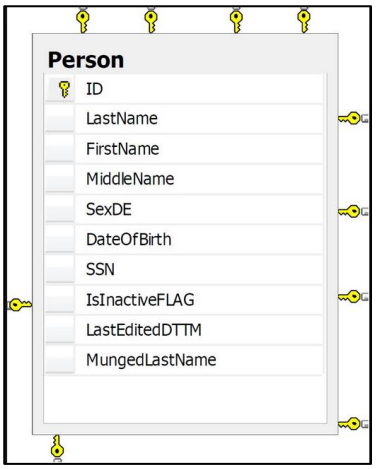
---

interrogating the contents of a computer as files and as machine-level data, Kivu is able to quickly target and assimilate a universe of potential PII/PHI. If Kivu relied on a single tool, there would be a higher likelihood of missing PII/PHI.

[7] PSPad is a text editing utility. ( Source: http://www.pspad.com/)

### i. PII/PHI Structure and Composition

The search for PII/PHI begins with knowledge of how PII/PHI may be documented and stored.




*In database tables, PII elements such as SSNs may be clearly identified, easily searched and correlated back to specific individuals. Databases can provide significant benefits in the tracking and management of*

Kivu confirmed that the EMR application database was secured using passwords, specific trusted user authentication (that was different than the access passwords), and user controls for the server hosting the database. Although the server hosting the database had been compromised by the attackers, Kivu determined the attackers had not managed to access the contents of the ERM application database itself[8]. Kivu was therefore able to focus the initial searching on data elements outside the EMR application.

Working outside the EMR database (but focusing on folders created on the host server by the EMR application or EMR software users), Kivu was tasked with searching thousands of files of different file formats and data types. Patient data could exist in multiple formats, such as EMR-generated reports, billing data tables, Excel data anlysis files, fax documents or patient letters.

Searching for PII/PHI was conducted through the recognition of patterns for certain types of PII/PHI such as SSNs and the patient numbering convention used by the specific Covered Entity[9]. Kivu maintains a library of pattterns for many types of PII/PHI, and each pattern has a specific set of rules that characterize the pattern (e.g., how many characters long or range of values in a pattern). For example, an SSN is defined as a nine digit numeric pattern in the format 000-00-0000. In computer systems, there are two common patterns that exist for storing SSNs (000-00-0000 and 000000000), but these patterns may overlap with other types of data such as medical

---

[8] While determining PHI/PII content of the database was outside the scope of Kivu's search, the database would have been an easily searchable source for PII/PHI. The knowledge required to locate PII/PHI in the database would have been defined by the tables and data elements present in the database. Reviewing the schema of the database would quickly determine data elements likely to contain PHI/ PII, and searches would also be carried out against the database itself for likely PII/PHI variants (e.g., all SSNs/ nine-digit patterns or specific searches for SSNs associated with specific individuals linked to the covered entity).

[9] Pattern recognition refers to the ability to identify a data element such as a credit card number, an object such as an image, or a relationship such as an individual and a listing of medical conditions. (Source: http://www.cs.rit.edu/~rlaz/prec20092/slides/Overview.pdf)

record numbers (MRNs)[10]. Data patterns for an SSN are filtered to exclude overlapping patterns with other data and are also refined with several rules that exclude certain SSN patterns (e.g., SSNs cannot start with 666 or 000)[11] Kivu examined the compromised server for SSN patterns, other frequently used PII sources (e.g., driver's license numbers and credit card numbers) and unique data patterns identfied from a review of the specific EMR system.

**Locating SSN Patterns**

| Nine-Digit SSN Patterns | Rules to Limit False Positives for Nine-Digit SSNs |
|---|---|
| • 333-00-4567<br>• 333004567<br>• 033 00 4567 | • xxx-xx-4567 (only last 4 digits SSN. Not complete.)<br>• xxxxx4567 (only last 4 digits SSN. Not complete.)<br>• 772-00-2365 (00 not used as sequence for 4th and 5th digits)<br>• 666-22-8756 (666 not valid first 3 digits)<br>• 415-55-1212 (mistyped phone number) |

Since computer data may be stored in different formats (or encoded), Kivu modified its search patterns to locate potentially hidden (but present) PII/PHI. Computers encode data in order to store, secure, process, transmit and present data. Encoding also assists computers in distingushing one type of data from another (such as computer binary 0s and 1s versus English language words). In the table below, the SSN pattern is displayed in various encoding formats that illustrate how an SSN could be missed if the search was limited to an ASCII character search (i.e., human language search).

[10] A medical record number (MRN) is used to identify and to associate medical information with an individual. (Source: http://policy.ucop.edu/doc/1100168/LegalMedicalRecord)

[11] For the nine-digit SSN pattern, there are 1 billion possible combinations. However, many SSN patterns are invalid due to rules established by the Social Security Administration ("SSA"). The available number of SSNs is closer to 750 million. (Source: http://www.ssa.gov/employer/stateweb.htm)

**Encoded SSNs**

| SSN Example | Encoding |
|---|---|
| 333-00-4567 | Unicode or ASCII[12]) |
| %33%33%33%2d%30%30%2d%34%35%36%37 | URL |
| MzMzLTAwLTQ1Njc= | Base64 |
| 3333332d30302d34353637 | Hexidecimal |
| 00110011 00110011 00110011 00101101 00110000 00110000 00101101 00110100 00110101 00110110 00110111 | Binary |

A final consideration is file type. When PII/PHI is stored in a user created document, there is usually a greater possibility of locating PII/PHI with data patterns. The document structure and formatting of common user created file types is well-known and widely recognized across many search-enabled software products (e.g., dtSearch[13]). The level of difficulty to search other file types increases when PII/PHI is located within: nested compressed files (e.g., ZIP file format — which must usually each be mounted, or opened, to be searched); compound files such as Microsoft Office documents; or file types associated with machine output (e.g., XML files[14]). Programmatic data file structures such as JSON[15] or XML file types may not be recognized by computer systems or may contain PII/PHI formatted with special characters (e.g., located within characters such as "<", end of line characters or non-breaking space characters) that result in skipping data patterns that should be responsive to search — such as a health plan name and insurance subscriber ID.

In the following illustration, example patient demographic data is outlined in JSON and XML file formats. The structure of these files present potential hurdles for the identification and classification of potential PII/PHI. For example, an SSN search pattern could hit within the MRN value as a false positive if the social security data pattern is not properly defined before search — the MRN contains nine consecutive numbers in its sequence. Alternatively, a search for the MRN data pattern could be missed if the search is constrained such as searching for MRN values that are delimited by space characters. The MRN values below are surrounded by extra characters and are not delimited by spaces. A search to identify this type of MRN should employ rules to discover the MRN pattern irrespective of characters that may encapsulate the MRN value.

[12] ASCII refers to a method using to identify English language characters, and several methods have been developed for the recognition of characters across multiple languages (e.g., Unicode). Computers use methods such as ASCII to present, store, process and transmit text. (Source: http://www.unicode.org/reports/tr17/#CharactersVsGlyphs)

[13] dtSearch creates search software for standalone search (e.g., desktop search) or enterprise search (e.g., servers, Internet database, etc.). (Source: http://dtsearch.com/)

[14] XML is an acronym for Extensible Markup Language. XML is a specification for storing data, and can be highly simplified such as a set of data elements in a report or elaborate such as a data store for documents. (Source: http://www.w3.org/TR/REC-xml/)

[15] JSON (JavaScript Object Notation) is referred to as a "lightweight data-interchange format" that results in easier data organization and translation by computer systems. (Source: http://www.json.org/)

**Programmatic File Types**

John Smith.json

```
{
  "Patient": {
    "MRN": "MS4938264957651209867 4",
    "Name": {
      "LastName": "Smith",
      "FirstName": "John",
      "Middle": "William"
    },
    "IntakeCriteria": {
      "DmDxDate": "2013/07/15",
      "InitialHgbA1c": "6.5",
      "CoMorbid": [ "Hypertension"]
    },
    "Labs": {
      "LDLCholestrol": {
        "LDLLevel": "126.3",
        "LDLResultDate": "2013/08/31"
      },
      "SerumCreatinine": {
        "CreatinineLevel": "1.4",
        "CreatinineResultDate": "13/09/01"
      }
    },
    "CareLocation": {
      "FacilityName": "Main Street Clinic",
      "ContactEmail": "doctor@mainstrclin.net"
    }
  }
}
```

John Smith.xml

```
<Patient>
      <MRN>MS4938264957651209867 4</MRN>
      <Name>
           <LastName>Smith</LastName>
           <FirstName>John</FirstName>
           <Middle>William</Middle>
      </Name>
      <IntakeCriteria>
           <DmDxDate>2013/07/15</DmDxDate>
           <InitialHgbA1c>6.5</InitialHgbA1c>
           <CoMorbid>Hypertension</CoMorbid>
       </IntakeCriteria>
       <Labs>
           <LDLCholestrol>
               <LDLLevel>126.3</LDLLevel>
               <LDLResultDate>2013/08/31</LDLResultDate>
           </LDLCholestrol>
           <SerumCreatinine>
               <CreatinineLevel>1.4</ CreatinineLevel >
               <CreatinineResultDate>13/09/01</CreatinineResultDate>
           </SerumCreatinine>
      </Labs>
      <CareLocation>
          <FacilityName>Main Street Clinic</FacilityName>
          <ContactEmail>doctor@mainstrclin.net</ContactEmail>
      </CareLocation>
</Patient>
```

The ability to search is dependent on a thorough understanding of source data, the use of pattern recognition to identify potential PII/PHI, and the ability to adjust data patterns for search that is agnostic to how data may be encoded and stored.

**ii. Search Rules Definition**

The search for data patterns occurs through search rules. A search rule defines the requirements to identify a pattern (such as XXX-XX-XXXX for an SSN) and any limitations to the search pattern (e.g., a date of birth patttern may exist on more than one line in a file).

**Transforming Data Patterns to a Search Rule**

| Example SSN Search Rule |
| --- |
| ^((?!000)(?!666)([0-6]\d{2}\|7[0-2][0-9]\|73[0-3]\|7[5-6][0-9]\|77[0-1]))(\s\|\-)((?!00)\d{2})(\s\|\-)((?!0000)\d{4})$ |

The illustration above displays a strictly defined search rule for an SSN pattern — a nine-digit numeric sequence with restrictions on the use of certain numeric values. The base rule for an SSN pattern is "\d{3}-\d{2}-\d{4}" (e.g., 000-00-0000). This search expression is expanded for restrictions on valid SSNs such as those that do not begin with "000" or "666". After the search expression is defined, it is tested and validated for use in search.

The application of search rules is limited to files containing machine-recognizable text. Files containing text stored in an image-based format (e.g., a PDF file of a scanned paper document) were outside the scope of PII/PHI enumeration for the EMR server. In cases where Kivu has encountered image-based content, Kivu has obtained limited success with the use of technologies such as Optical Character Recognition (OCR) to achieve computer-recognizable

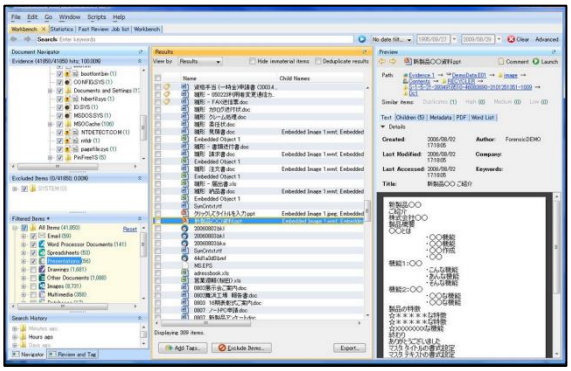
text[16] from image files. However, in cases of text stored in an image-based format, Kivu has recommended the use of sampling and manual review protocols to accurately identify and validate PII/PHI enumeration.

### iii. Search Tools

Kivu employed a combination of search tools to locate PII/PHI on the compromised server using a clearly defined set of search rules (as discussed above). The selection of search tools for a given matter depends on several factors such as file type and structure, data structure, data format(s), encoding, and volume of data.

From the perspective of PHI/PII analysis, search tool capability can be divided into human language search and data pattern (or machine data) search. Specifically, the distinction resides in the ability to implement data pattern search. This nuance is especially important for encoded data or data that is wrapped in "code" (e.g., software code or XML tags) which has a high probability of being "overlooked" by many search tools since "code" is often excluded from human-language search technology.

For file types such as email and Microsoft Office files which are rich sources of human language content, Kivu may use indexing tools[17] such as dtSearch or Nuix . Both tools work well for human language content search and have the capacity to locate data patterns within such documents. If there are thousands of files, a tool such as Nuix is also highly customizable and has the capability to expediently search a wide range of file types.



**Example NUIX Screen Capture**

In the present case, Kivu processed all user created files related to the EMR application and found on the compromised server through an indexing tool. This allowed for a rapid identification of PII/PHI and the ability to confirm that certain types of common user files, created by the EMR application, did not contain PII/PHI.

---

[16] In cases where such files are encountered, technologies such as Optical Character Recognition (OCR) may be used for text recognition. (Source: https://www.princeton.edu/~achaney/tmve/wiki100k/docs/Optical_character_recognition.html)

[17] An index is an inventory of words for a repository of files created by a search engine for the purpose of quickly locating content. (Source: http://support.dtsearch.com/webhelp/dtsearch/default.htm#creating_an_index.htm)

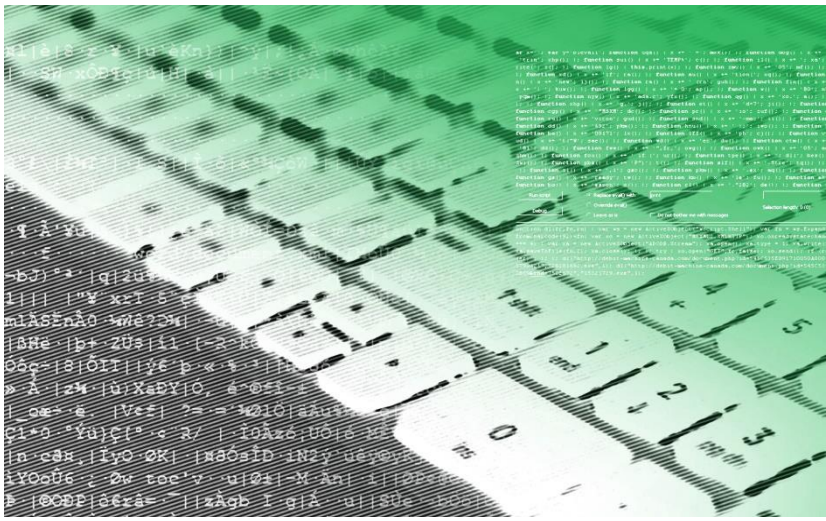
As a search progresses, Kivu may employ tools such as EnCase or Powergrep to examine files at the data level when file types such as server logs do not respond well to human-language-based search. Data may also be stored in proprietary file formats that many search tools cannot penetrate[18]. In some cases, Kivu may need to develop its own programming script written in languages such as python, perl, or awk to locate PII/PHI[19].

### iv. Data Source Profiling

In the case of the compromised server hosting the EMR database, Kivu quickly identified files containing PHI/PII, and that the files were resident in the file system outside the EMR database. Kivu reviewed directory paths associated with the externally stored files to ascertain possible explanations for the discovered files. Analysis indicated that the files were stored in directory locations specifically set up to organize files for patient management, medical practice-related communications, and patient data archiving. Storage locations included:

a. **EMR Software and Database Directories.** Programmatic files associated with the EMR software, the EMR database and database transaction log are stored in pre-defined directory locations. The locations are pre-configured within the EMR software (i.e., default locations where files created by the application are stored – and presumably accessed if a query is made to the database). Within these directories, Kivu found a database transaction log that was not secured by the EMR application. The log contained a large amount of patient data (e.g., name and patient record number and clinical notes) that should have been deleted after the patient data was entered into the EMR database — a reconciliation of the log file to its parent database. Presence of the patient data within the log suggested a corruption in the reconciliation process and clearance of patient data from the log.

b. **Other EMR Directories for Externally Stored File-based Records.** While databases may be efficient at storing highly structured data elements such as patient demographics (e.g., names and dates of birth), it is not uncommon for databases to function poorly as repositories for individual files (e.g., lab results or x-rays). As a result, certain forms of patient information such as HIPAA acknowledgements, PDF files of patient communications, lab results, etc. are likely to be stored externally from the EMR database and are not protected by the secure EMR application.



[18] Refer to Kivu's whitepaper, "Approach to E-Discovery Boolean Search", for a discussion factors that influence the use of search tools. (Source: http://kivuconsulting.com/wp-content/uploads/2012/08/Kivu_Boolean_Searching_Guide.pdf)

[19] The University of Tennessee provides an overview of Python and programmatic search of data patterns. (Source: http://web.eecs.utk.edu/~bvz/teaching/cs465Fa11/notes/Python/)

Review of external file storage

Many of these files were stored in directory paths consistent with the use of a comprehensive EMR software product. The illustration below presents an example of the directory structure Kivu observed for fax documents stored by the EMR application. The faxes contained date-based naming conventions — an indicator of an automated file-naming convention — and were stored in date-named directories. The files contained physician recommendations and other confidential patient information. Kivu compared samples of the faxes to patient records maintained by the secure EMR application to corroborate the relationship of the files to EMR patient data. Additionally, Kivu identified the presence of an active fax line connected to the server, establishing the possibility of fax-based transmissions.

**Example File Location for Files Containing PII**


The difference in security for externally stored files versus an EMR application

Security requirements for all EMR-related patient files should be the same, whether stored within the EMR application, or in an external location. The actual security controls however may be different. For example, access to files with patient lab results that reside on a Windows server outside the EMR application may be restricted through server-level access controls and BitLocker encryption. The EMR application may interact with these files when a medical professional is reviewing lab results, but the EMR software does not control file storage. In this scenario, file-level security may become the responsibility of IT/InfoSec staff and a point of potential security breakdown. Unless IT staff understand the respective security requirements for both the EMR application and its related externally stored files, it is highly probable that the externally stored files containing PII/PHI could be stored with minimal or no security.

The presence of unusual externally stored files

In addition to the externally stored patient files described above, Kivu found daily patient medical and appointment records. These appeared to be secondary records[20] created by the EMR software:

- These files were named with a date-derived naming convention (e.g., 20150115_FileName.txt) and stored within a directory named “archive”;
- Some of the files contained field names such as “firstname” — indicative of reporting content from a structured database;
- Each file contained one or more line-item entries, and each entry had characteristics of EMR database activity such as:

[20] A “secondary record” is a related set of database records that support the EMR database with functions such as data verification.

- Date and time stamp for each entry;
- An identified user name and workstation name associated with use of the EMR application;
- Patient name, SSN, patient id, date of birth, or appointment notes; or
- Health insurance plan and subscriber detail.

In Kivu's experience, one may find files of extracted EMR database content such as a data table of patient demographics (e.g., patient id, name, gender, etc.) on the same server that hosts an EMR database. Such files may be used by a medical clinic to switch EMR applications or to perform specialized analysis of EMR data (e.g., assessment of clinical quality measures such as patient engagement[21]). However, it was unusual to find hundreds of files that collectively had the hallmarks of a patient records system where the patient records in the text files mirrored daily activity of the EMR application's secure database. These text files could be designed for the particular application for re-building a corrupt EMR database or providing a method to audit patient activity. Whatever the purpose, the unsecured text files indicate a poorly conceived concept from the EMR software developers. The quantity of files and structure of content within the files would not have been created by an end user such as a front-desk assistant or a physician.

c. **User-created Directories.** Users of EMR software may store EMR-originating data separately from a system's official storage location (i.e., outside a secure application). This may occur in scenarios such as a patient requiring customized medical intervention, management of billing issues, or research studies using patient records. Typically, application users create their own directories and maintain their own files on their workstations or network shares. In this case, IT staff should have knowledge of these practices and should have controls in place. As an example of the above, search of the compromised server in this case resulted in the identification of patient data in multiple directory shares created by individual physicians, with little apparent structure or controls, e.g.:
    a. The location of the shares — some were off the root of the main directory, some were sub-shares to named physician directories;
    b. Different naming conventions for the shares; and
    c. Divergent date ranges and content of the patient data stored in these shares.

[21] Patient and Family Engagement is one of six National Quality Strategy ("NQS") domains established by the Centers for Medicaid and Medicare Services ("CMS") for the measurement of health care quality management. (Source: http://www.cms.gov/Regulations-and-Guidance/Legislation/EHRIncentivePrograms/2014_ClinicalQualityMeasures.html)

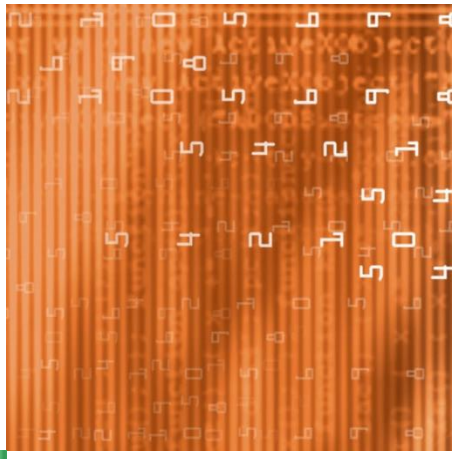
## Key Findings

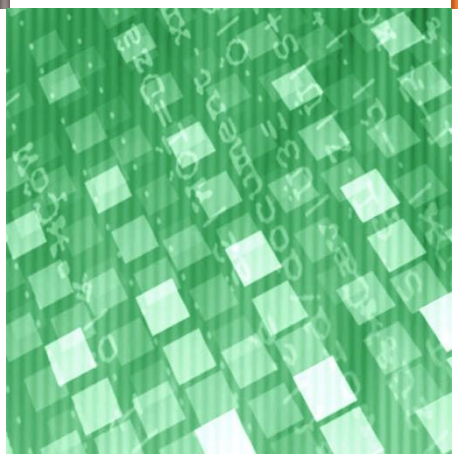
**Database transaction logs associated with the EMR database stored a significant amount of identifable PII/PHI that lacked encoding or other obfuscation sufficient to deter PII/PHI detection.**

The database transaction logs associated with the EMR software contained searchable text, and PII/PHI were located by server-wide searches. While Kivu employed sophisticated tools to complete PII/PHI analysis, the EMR's database transaction log could also be searched using a text editor such as PSPad. The directory containing these transaction logs had no limitations on user access, leaving PII/PHI in these logs vulnerable to rogue employees or attackers who gained access to the server.

**Extensive patient information including PII and PHI was stored in text archive files and other EMR-related files outside the secure EMR database.**

- Kivu found thousands of text files that reported years of detailed database activity in clear text, apparently designed to assist in upgrading the EMR software or creating an audit trail. The files were stored within directories related to the EMR software but outside the secure EMR database.
    - The files identified EMR software users for patient-specific transactions at specific dates and times. Database sources were identified with respective data element names and patient data.
    - An additional concern was the presence of usernames and passwords related to the EMR application. Passwords were clear-text (or searchable text) and were not obscured by encoding or encryption. Since the user names and passwords were stored in database logs, this would appear to be an oversight of poor application and database design. As a best practice, login credentials are not stored in the same files or database tables as patient data. While a review of the passwords indicated that they were all non-active passwords, the possibilty of "leaking" current passwords appeared a major concern.
- Numerous patient files used or created by the EMR application had been stored outside the database. Files included faxes, perscription information, drivers' licenses, insurance information, billing information, and user-created reports.





## Conclusions

The EMR software and the server housing the EMR software lacked appropriate controls to secure PII/PHI. The presence of EMR login credentials in text-searchable files potentially negated the use of encryption for the EMR database. Unsecured directories provided the opportunity for any user to browse the server and potentially locate files containing patient data.

The audit capabilities of the EMR software were limited to the EMR database. As a result, externally stored files with patient data were outside the reach of the EMR software. PII and PHI could have been exfiltrated without leaving evidence of file activity. For example, on a Windows computer, a hacker could use a Robocopy[22] command to copy files, and use of this command would leave no evidence of file access.

Using sophisticated search tools employing data pattern recognition, Kivu was able to identify numerous instances of PII and PHI on the compromised server. The client was surprised by the result, as they believed the EMR system was secure and HIPPA compliant. This was a painful lesson in the numerous (and dangerous) ways that sensitive data can *leak* from an otherwise secure system.

[22] Robocopy or "Robust File Copy" is a command-line utility integrated into the Windows operating system. Robocopy provides the ability to copy one or more files to different locations on the same computer or to other locations in a network environment. (Source: https://technet.microsoft.com/en-us/library/cc733145.aspx)

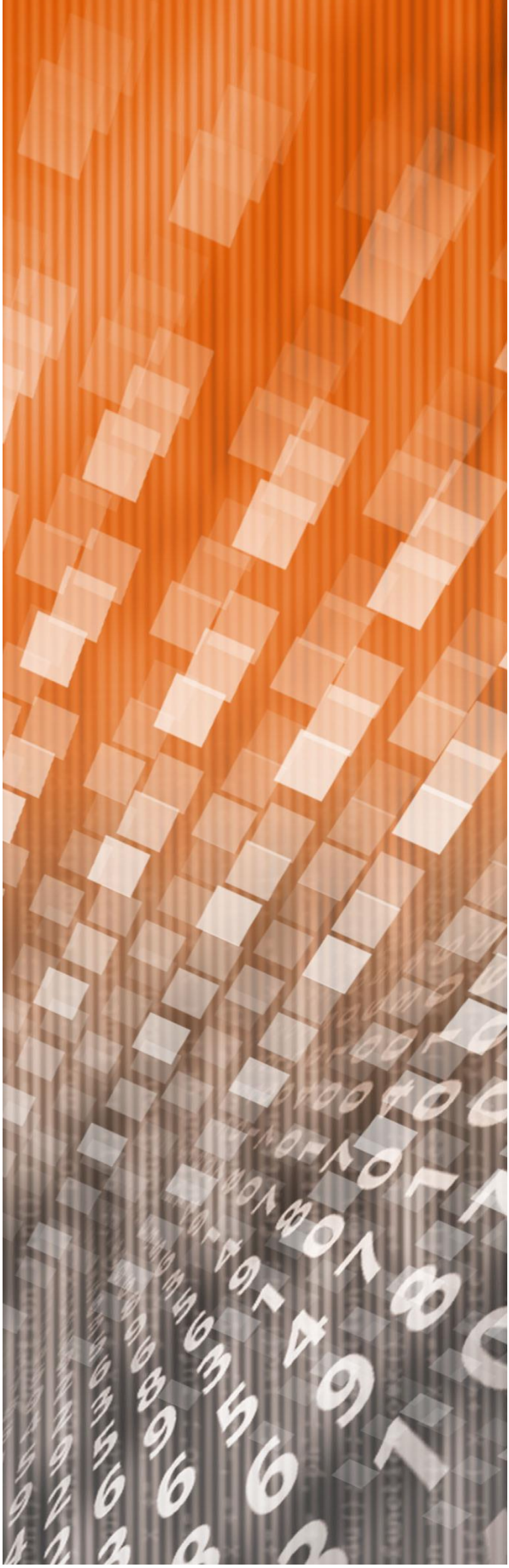
## Legal Conclusions

The misnomer of HIPAA compliant software is prevalent in the health care industry. Too often, HIPAA-regulated entities rely on vendor controls and claims of compliance as a substitute for their own HIPAA security programs. While the software itself may meet the requirements for HIPAA compliance for the discrete functions it performs, the truth of the matter is that no software or system that handles PHI is HIPAA compliant until it has undergone a risk assessment by the regulated entity to determine its functionality and efficacy of its security controls in the user's environment. While HIPAA itself is an unarticulated framework, adherence to its required risk management processes and industry-best practices should protect organizations from attacks similar to the one described above.

HIPAA requires that both covered entities and business associates maintain a security management process to implement policies and procedures to prevent, detect, contain, and correct security violations.[23] The foundational step in the security management process is the risk assessment, which requires regulated entities to conduct an accurate and thorough assessment of the potential risks and vulnerabilities to the confidentiality, integrity, and availability of electronic protected health information held by the entity.[24]

NIST Special Publication 800-66 identifies a protocol organizations may use for conducting a HIPAA compliant risk assessment. 800-66 generally identifies nine steps an organization should take in this regard. Significantly, the first two steps of the risk assessment process should be read together to identify all information systems containing PHI and ensure that all PHI created, maintained, or transmitted by the system is being maintained appropriately and security controls are applied.

In the context of third party software and systems, the risk assessment process should be used to identify hidden repositories of PHI where unintended business functions or improper implementation cause PHI to be located outside of an organization's secure environment. If third party software and systems are not identified within the scope of a risk assessment, and a disclosure or audit occurs, the government may impose penalties for not



[23] 45 C.F.R. § 164.308(a)(1)(i).
[24] 45 C.F.R. § 164.308(a)(1)(ii).

conducting a thorough risk assessment. Additionally, there is potential for third party lawsuits if a disclosure results. In a data breach dispute, the argument usually boils down to whether the controls the organization had in place were reasonable to protect PHI. In many cases, the plaintiffs use HIPAA as a standard of care, so that if an organization was not in compliance, the plaintiffs will argue the organization did not take reasonable steps to protect PHI.

While not conducting an accurate and thorough risk assessment may result in regulatory enforcement or litigation risk, failing to identify hidden repositories of PHI may also result in other HIPAA violations. If data is stored outside of its intended repository, it is unlikely that an appropriate data classification and associated security controls have been applied to the hidden repository. The result is that it is unlikely the HIPAA regulated entity is meeting the required technical implementation specifications of the HIPAA Security Rule with regard to the information contained in the hidden repository. For example, in such situations it is unlikely that an organization has appropriate access[25] and audit[26] controls in place on systems that are not intended to store PHI.

Reliance on claims of HIPAA compliance with regard to third party systems and software poses a real risk of government enforcement and private lawsuits should a disclosure or audit occur. If such systems have not undergone a thorough risk assessment in an organization's environment, such systems may create hidden repositories of PHI. In this regard, the government may impose penalties for not conducting a risk assessment. Additionally, the PHI that resides in such hidden repositories is unlikely to have appropriate security controls applied, bringing the organization out of compliance with the HIPAA Security Rule's technical implementation mandates.

[25] 45 C.F.R. § 164.312(a)(1).

[26] 45 C.F.R. § 164.312(b).

# Appendix

## Common Vulnerabilities in EMR Software

Software is developed for a specific purpose such as managing patient information or insurance billing. Software's core functionality is created during a development cycle, and security may be incorporated into the development process or may be an afterthought. However, security is optimized when it exists wthin a software application and the environment where the application is hosted.

1. **At the device level where the software is installed, software integrates with its host operating system, file system and network environment. The intersection between an application and its host environment could create significant PII/PHI exposure risk.** For example, PostreSQL databases in a Linux environment can be controlled through the operating system. A user or a hacker could access the content of a PostreSQL database without ever using the database. Software applications require security reinforcement at the file, directory and server level to protect PII/PHI-containing files stored outside secured database environments.

2. **Software, particularly database software, is often vulnerable due to poor security upgrade practices and loose configurations.** Secure software and data are maintained through an ongoing process of security checks, updates and re-configurations. This includes determining the frequency with which a software distributor provides updates, bug fixes and timely warnings about problems discovered with the software. Since databases are often customized, configurations and custom database features should be periodically examined for security issues. An ongoing program of updates based on research and testing should be implemented to maintain a consistent level of security.

3. **Even when security features are established, users may change security features to appease users or to simplify IT tasks.** It is not uncommon for security features such as restricted folder permissions to be manually removed as a solution for quick access to a specific folder's resources. Reducing security for ease of use may also be observed in the re-configuration of software. For example, Microsoft removed a function known as "xp-cmdshell" from its SQL server beginning with version SQL Server 2005 since "xp-cmdshell" allowed server-level access and control from. Although Microsoft removed the "xp-cmdshell" feature, Microsoft provided a means to re-instate "xp-cmdshell". If a developer or system administrator implements "xp-cmdshell" without the appropriate controls in place, the respective SQL database could be used to manipulate the server.

4. **Delayed software upgrades or improper upgrade installation may increase the potential for compromise.** The software upgrade process is often the responsibility of the organization purchasing the software. The scope, cost and complexity of upgrades influence an organization's decision on whether to upgrade or stay with an existing software version. In many cases, software upgrades substantially lag behind the rollout of upgrades in operating systems and Internet browsers. If a software product ties an organization to older technologies such as the Microsoft Windows XP operating system, an organization may face increased risk of potential attack due to the larger number of identified vulnerabilities and the lack of patching to correct vulnerabilities.

5. **External communication channels are often incorporated into software applications to enable functionality such as transmitting faxes/emails, or to allow for access by outside administrative support. These communication channels are often left unsecured with default configuration settings and administrative credentials.** Hackers use default administrative credentials (easily discoverable by Internet research) to test servers that are accessible on the Internet. Default credentials are set up using automated software to scan and search for vulnerable systems, with brute-force password guessing yielding server-level access. Immediately changing all default passwords upon initial implementation, periodic evaluation of firewall logs, requiring strong passwords, and whitelisting access for approved individuals are important steps in safeguarding exposed communication channels.

6. **Audit logs are typically developed to support a specific software application, but use of audit logs may be disabled or ignored.** Audit logs that are designed for a specific application are usually designed to enhance application performance, not improve security. However, they can be often be configured to allow an organization to review for unauthorized or unusual access. This does require an internal commitment to provide sufficient resources to review the generated logs or, as a minimum, establish guidelines and resources to store the logs for use in the event of an audit. This may require storing logs for 3 - 6 months or longer, in a readily accessible format. A storage (and recovery/ review) protocol should be included in the organization's incident response plan.

## About Epstein Becker Green

Epstein Becker & Green, P.C., (www.ebglaw.com) is a national law firm with a primary focus on health care and life sciences; employment, labor, and workforce management; and litigation and business disputes. Founded in 1973 as an industry-focused firm, Epstein Becker Green has decades of experience serving clients in health care, financial services, retail, hospitality, and technology, among other industries, representing entities from startups to Fortune 100 companies. Operating in offices throughout the U.S. and supporting clients in the U.S. and abroad, the firm's attorneys are committed to uncompromising client service and legal excellence.

## About Kivu

Kivu (www.kivuconsulting.com) is a nationwide technology firm specializing in the forensic response to data breaches and proactive IT security compliance. Headquartered in San Francisco with offices in Los Angeles, New York and Washington DC, Kivu handles assignments throughout the US and is a pre-approved cyber forensics vendor for leading North American insurance carriers. Kivu's forensic investigators are experienced in protecting organizations against compromise of data, theft of trade secrets and unauthorized access to data. Our qualifications include forensic certifications (Encase Certified EnCE, SANS GCIH Incident Handlers, Certified Ethical Hackers, and reverse malware experts); IT certifications (Certified Information Systems Security Professional CISSP and Certified Information Systems Auditor CISA); and prior backgrounds as legal counsel, IT administration, and network security. Kivu's investigators have testified as computer forensic experts in state and federal court, and presented their findings to state and federal regulators.

## Contact Information


**Megan Bell, CIPT, GWAPT**
Senior Director, Analytics
Kivu – San Francisco

T: (415) 524-7327
E: mbell@kivuconsulting.com

**Adam C. Solander**
Member of the Firm
Epstein Becker Green– Washington, DC

T: (202-861-1884
E: asolander@ebglaw.com


EPSTEIN BECKER GREEN

KIVU